2.13 Verification Results for Multipath and Diffraction

In general, the multipath and diffraction FE was found to be implemented as specified in Section 2.13 of ASP II; however, some discrepancies were found.

The broadest discrepancy is that ALARM 3.0 does not exactly match the MIT Lincoln Laboratory SEKE code; the following three items were different: (1) the constants in the logic to determine which propagation effects will be calculated, (2) the definition of the terrain profile, and (3) the extent of the tangent plane used in the calculation of the multipath effect. Apart from the differences from SEKE, several minor discrepancies were found in subroutines MLTPTH, VISBLE and SEKINT.

The quality of the code for the Multipath and Diffraction FE in ALARM 3.0 is generally good; however, subroutines LAPROP, FIRST, SECOND, PROFIL, VISBLE, MLTPTH, and SAVRST should be considered for improvement.

Internal documentation is generally good, but some minor deficiencies were found. The ALARM external documentation for this FE is inadequate, because the analyst's manual gives an unacceptably incomplete description of the relevant algorithms.

The tables listed below summarize the desk-checking and software testing verification results for each design element in the Multipath and Diffraction Functional Element. One entry appears for each design element or supporting element that was examined. The results columns contain checks if no discrepancies were found. Where discrepancies were found, the desk check results column contains references to discrepancies listed in table 2.13-12, while the test case results column lists the number of the relevant test case in tables 2.13-15 through 2.13-19. More detailed information on the results is recorded in those tables.

Table 2.13-1 Verification Matrix for Decision Logic

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-1 Propagation Effects Decision Logic	LAPROP all	D1	13-4 through 13-10	✓
13-2 Clearance Ratios	FIRST all	✓	13-18	✓
13-3 Diffraction Effects Ratio	SECOND all	✓	13-19,20	✓
13-4 Factors for Special cases	LAPROP 249-299 and 392-432	✓	13-11 through 13-15	✓

Table 2.13-1 Verification Matrix for Decision Logic (Continued)

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-5 Special Cases, No Propagation Effects	LAPROP 488-499	✓	13-1,2,3	✓
13-6 Application of Propagation Effects Factor	LAPROP 473-474	✓	13-4 through 13-15	✓
13-7 Terrain Profile	PROFIL all SEKINT 150-151	D2	13-21,59,62	✓
13-8 Terrain Visibility	VISBLE all SEKINT 191-203	D3, D4	13-22,23,24,59,62	✓

Table 2.13-2 Verification Matrix for Specular Multipath

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-9 Multipath Effects Factor	MLTPTH all	D5	13-25,26,27,30,31,32,34	✓
13-10 Reflection Coefficient	RFLECT all SEKINT 153-167, 187	✓	13-35,59,62	✓
13-11 Terrain Roughness Factor	MLTPTH 588 SEKINT 176-178	D6	13-28,59	✓
13-12 Width of Fresnel Zone	MLTPTH 516-527	D7, D8	13-26	13-26
13-13 Specular Reflection Points	MLTPTH 186-226	D9, D10, D11	13-30	13-30

Table 2.13-3 Verification Matrix for Knife Edge Diffraction

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-14 Locate Knife Edges	KEDIFF all	✓	13-36,37,38	✓
13-15 Knife Edge Clearances	DEYGOU 235-238 260-281	✓	13-43	✓
13-16 Individual Knife Edge Diffraction Factors	DEYGOU 213-214	✓	13-41	✓
13-17 Combined Knife Edge Diffraction Factor	DEYGOU 386-403	✓	13 -39,40,42	✓

Table 2.13-3 Verification Matrix for Knife Edge Diffraction (Continued)

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-18 Fresnel Sine and Cosine Integrals	FRESNL all	✓	13-45,46,47	✓

Table 2.13-4 Verification Matrix for Spherical Earth Diffraction

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-19 Spherical Earth Diffraction Factor	SEDIFF 247-320	✓	13-48	✓
13-20 Normalized Ground Range and Heights	SEDIFF 209-255	✓	13-49	✓
13-21 Airy Region Determination	AIRY all	✓	13-50	✓
13-22 Airy Function in Connection Region	CONECT all	✓	13-51	✓
13-23 Airy Function in Integral Region	GAUSSQ all	✓	13-52	✓
13-24 Airy Function in Power Series Region	POWERS all	✓	13-53	✓

Table 2.13-5 Verification Matrix for Initialization and Utilities

DESIGN ELEMENT	CODE LOCATION	DESK CHECK RESULT	TEST CASE ID	TEST CASE RESULT
13-25 Linear Fit	LINFIT all	✓	13-54	✓
13-26 Parabolic Fit	PARFIT all	✓	13-55,56,57	✓
Input	SEKINP all	✓	13-59,61	✓
Error Checks	SEKERR all	✓	13-59,60	✓
Echo Inputs	SEKPRT all	✓	13-63	✓
Radar/Jammer Data Exchange	SAVRST all	✓	13-58	✓

2.13.1 Overview

Version 3.0 of ALARM models specular multipath as well as spherical and knife edge diffraction. Diffuse multipath is not treated. Specular multipath is an interference condition that occurs when reflected energy is received from round-trip paths other than the direct path to and from the target. The power, direction and phase of the interfering signals are determined by the radar range equation, path geometry and the surface properties of the terrain. Diffraction is another interference effect that modifies the propagation of the wave front. Knife edge diffraction is due to partial blockage of the transmitted energy by discrete terrain features. Spherical earth diffraction is due to continuous wave interference over a gently curved surface.

ALARM uses four primary subroutines to implement the multipath and diffraction FE:

1. LAPROP is the controlling routine that sets up the problem and calls the other routines depending on the geometry, terrain and user choices.

2. MLTPTH calculates the specular multipath component.

3. KEDIFF calculates the knife edge diffraction factor.

4. SEDIFF calculates the spherical earth diffraction factor.

Several supplementary and utility routines that are called from the four primary routines also are incorporated into this FE. All subroutines implementing this FE are listed below in table 2.13-6.

Table 2.13-6 Subroutine Descriptions

MODULE NAME	DESCRIPTION
AIRY	Performs an Airy function evaluation
CONECT	Determines the value of an Airy function in the "connection" region of the complex plane (region 1)
DEYGOU	Finds knife edges and calculates knife edge diffraction
FIRST	Determines minimum ratio of clearance of direct ray to Fresnel clearance at each point in terrain profile
FRESNL	Determines Fresnel sine and cosine integrals
GAUSSQ	Determines the value of an Airy function in the "integral representation" region of the complex plane (region 2)
KEDIFF	Determines knife edge diffraction loss
LAPROP	Determines the correct combination of multipath and diffraction effects

Table 2.13-6 Subroutine Descriptions (Continued)

MODULE NAME	DESCRIPTION
LINFIT	Calculates a linear fit
MLTPTH	Calculates the multipath contribution to the propagation factor
PARFIT	Calculates a parabolic fit
POWERS	Determines the value of an Airy function in the "power series" region of the complex plane (region 3)
PROFIL	Builds a terrain profile for the ground trace of the ray from the site to the target (or jammer)
RFLECT	Determines the complex reflection coefficient of a plane earth
SAVRST	Preserves/restores radar-to-target terrain profile common variables to allow use of common multipath/diffraction logic for stand-off jammers
SECOND	Determines the ratio of highest mask to Fresnel clearance at that point
SEDIFF	Determines the spherical earth diffraction loss
SEKERR	Confirms the limits on user input data for multipath/diffraction
SEKINP	Reads user input data for multipath/diffraction
SEKINT	Initializes internal variables for multipath/diffraction
SEKPRT	Prints user input data for multipath/diffraction
VISUAL	Sets up the terrain profile, determines if each point is visible from the radar

2.13.2 Verification Design Elements

The design elements defined for the multipath and diffraction FE are listed in tables 2.13-7 through 2.13-11. A design element is an algorithm that represents a specific component of the FE design. They are fully described in the Design Approach section of ASP II for ALARM 3.0. The first tables contain the design elements in the four major areas: decision logic, multipath, knife edge diffraction, and spherical earth diffraction. The final table contains design elements related to initialization and mathematical utilities.

Table 2.13-7 Decision Logic Design Elements

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
LAPROP	13-1 Propagation Effects Decision Logic	Top level decision algorithms for which propagation effects to use and how to combine them to obtain F.
FIRST	13-2 Clearance Ratios	Calculate clearance ratios for use in multipath/ diffraction decision logic

Table 2.13-7 Decision Logic Design Elements (Continued)

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
SECOND	13-3 Diffraction Effects Factor	Calculate h_m and Δ_0 for use in determining type of diffraction
LAPROP	13-4 Factors for Special Cases	Treat cases where Spherical Earth Diffraction is called for, but the series solution does not converge
LAPROP	13-5 Special Cases, No Propagation Effects	Set F to antenna gain (only) if the target is closer than terrain resolution or if user selects no propagation effects
LAPROP	13-6 Application of Propagation Effects Factor	Raise F to fourth power for target, second power for standoff jamming signal
PROFIL and SEKINT	13-7 Terrain Profile	Determine points in terrain profile between the radar and target (Geometry constants found in SEKINT)
VISBLE and SEKINT	13-8 Terrain Visibility	Find all points in the terrain profile that are visible to the radar (Geometry constants found in SEKINT)

Table 2.13-8 Specular Multipath Design Elements

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
MLTPTH	13-9 Multipath Effects Factor	Calculate factor for propagation effects due to specular multipath (F_m)
RFLECT and SEKINT	13-10 Reflection Coefficient	Calculate the reflection coefficient for a smooth plane
MLTPTH and SEKINT	13-11 Terrain Roughness Factor	Calculate roughness factor for specular multipath over water
MLTPTH	13-12 Width of Fresnel Zone	Calculate width of first Fresnel zone for each specular point
MLTPTH	13-13 Specular Reflection Points	Determine which points in the terrain profile are associated with specular reflection points

.

Table 2.13-9 Knife Edge Diffraction Design Elements

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
KEDIFF	13-14 Locate Knife Edges	Find all local maxima in terrain profile, then find those with smallest clearance ratios
DEYGOU	13-15 Knife Edge Clearances	Calculate Fresnel clearances for secondary knife edge diffraction points

Table 2.13-9 Knife Edge Diffraction Design Elements (Continued)

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
DEYGOU	13-16 Individual Knife Edge Diffraction Factors	Calculate factor for each knife edge
DEYGOU	13-17 Combined Knife Edge Diffraction Factor	Calculate factor for knife edge diffraction effects due to all knife edges and antenna gain
FRESNL	13-18 Fresnel Sine and Cosine Integrals	Compute numerical approximation to Fresnel sine and cosine integrals

Table 2.13-10 Spherical Earth Diffraction Design Elements

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
SEDIFF	13-19 Spherical Earth Diffraction Factor	Calculate factor for propagation effects due to spherical earth diffraction and gain (F_s)
SEDIFF	13-20 Normalized Ground Range and Heights	Calculate normalized ground range, antenna height, and target height
AIRY	13-21 Airy Region Determination	Determine formula to be used for $\overline{Ai}(z)$ based on region of complex plan containing z
CONECT	13-22 Airy Function in Connection Region	Calculate $\overline{Ai}(z)$ for z in the connection region
GAUSSQ	13-23 Airy Function in Integral Region	Calculate $\overline{Ai}(z)$ for z in the integral region
POWERS	13-24 Airy Function in Power Series Region	Calculate $\overline{Ai}(z)$ for z in the power series region

Table 2.13-11 Initialization and Utility Routine Design Elements

SUBROUTINE	DESIGN ELEMENT	DESCRIPTION
LINFIT	13-25 Linear Fit	Determine the least squares linear fit to a set of (x,z) data points
PARFIT	13-26 Parabolic Fit	Determine the least-squares parabolic fit to a set of (x,z) data points
SEKINP	Input	Read user inputs in DATASEKE
SEKERR	Error Checks	Check user inputs in DATASEKE to insure they are within appropriate limits
SEKPRT	Echo Inputs	Print values input in DATASEKE
SAVRST	Radar/Jammer Data Exchange	Interchange radar and jammer parameters before or after calculating propagation factor for stand-off jammer

2.13.3 Desk Checking Activities and Results

The code implementing this FE was manually examined using the procedures described in Section 2.0 of this report. Discrepancies discovered are described in the following tables.

.

Table 2.13-12 Code Discrepancies

DESIGN ELEMENT	DESK CHECK RESULT
13-1 Propagation Effects Decision Logic	D1: The constants in the decision logic do not exactly match those in the reference or in the SEKE code (ALARM uses .75 where SEKE uses 1.0). The developer explained this was done to make output match test results after correcting an error in the SEKE terrain profile logic.
13-7 Terrain Profile	D2: DELTAG is currently hardwired to the value 29.5π meters. This is correct for the DMA terrain data now used, but DELTAG must be reset if the database resolution is changed.
13-8 Terrain Visibility	D3: The terrain profile is defined differently than in the MIT SEKE code. D4: ATAN2 (x,y) is undefined if both x and y are 0. Theoretically, this could happen at line 176 if SIAZIN=0 (COAZIN=1) and $\beta_i=\pi/2-\Phi s$ (where β_i is the angle from the radar to the center of the earth to the i^{th} terrain point and Φs= latitude of radar site). (This could not be forced to occur during testing, so it is probably a rare problem.)
13-9 Multipath Effects Factor	D5: The gain in the direction of the specular point is based only on the transmit gain, ignoring the possibility that the receive gain is different.
13-11 Terrain Roughness Factor	D6: In the case where the terrain is water, the code and ASP II equation (2.13-12 c) do not agree with [A.1-5]. The 1/4 in the ASP II and in the code should be replaced by $\frac{1}{\sqrt{2\pi}}$. (However, testing showed that this discrepancy had negligible effect on the final value of ρ_s.)
13-12 Width of Fresnel Zone	D7: For the special case where the only specular point lies between the radar and the first terrain point, the zone width is never calculated, always set to zero. This results in no effect due to multipath. D8: The extent of the tangent plane at a specular point is calculated differently than in the MIT SEKE code. This is the subject of an unresolved MDR.
13-13 Specular Reflection Points	D9: The first terrain interval is treated as a special case, and is used only if NO other specular points are found. D10: The case where a specular point is exactly on a terrain point is not considered. D11: There is no protection against division by zero in the case of PSLOPE = 0.0, 1 + PSLOPE*TKNEPP(I)=0, or 1 - PSLOPE*TANGAM = 0.

Except as noted in table 2.13-13 below, overall code quality and internal documentation were evaluated as good. Subroutine I/O and logical flow were found to match the ASP II descriptions.

.

Table 2.13-13 Code Quality and Internal Documentation Results

SUBROUTINE	CODE QUALITY	INTERNAL DOCUMENTATION
LAPROP	Calls to several subroutines are implemented in more than one place in code, so perhaps the logic could be simplified; however, it is much simpler than the MIT SEKE code and overall code quality is good.	No purpose is given in the header. The list of inputs given in the header omits some inputs and lists some variables (NPROFL, TANMAX) which are generated by subroutines called by this routine.
SECOND	The variable CONST in this routine represents the value of Δ_0, the Fresnel clearance at the point where Δ/Δ_0 has minimum value. The value of Δ_0 has already been implicitly computed in subroutine FIRST and could be saved in a common rather than be recalculated. Furthermore, different variables are used for the calculation in the two subroutines. (Testing showed the differences between the two values obtained are insignificant.)	The header lacks description of purpose and output variables. Some input variables are also missing.
PROFIL	Code structure could be improved by using a conditional check to avoid calling VISBLE if NPROFL ≤ 0.	Some inputs are not defined in the header.
VISBLE	OK	Header lacks definition of some input variables. Also, "respectively" should be deleted from the definitions of COAZIN and SIAZIN.
MLTPTH	TANTRG is just a recalculation of SLPTGL. Why not use SLPTGL?	Subroutine purpose and may input variables are missing from the header. SLOPEL and SLOPER are misidentified in code comments at lines 378-390. The comment at records 485-490 would be clearer if the words "for use in determining width of first Fresnel zone" were added at the end.
KEDIFF	OK	Header lacks discussion of purpose and omits some inputs
DEYGOU	ILEFT, IMAIN, and IRIGHT are not checked for values greater than 2049, but they are protected in the calling routine (KEDIFF), so array bounds should not be violated.	Header lacks discussion of purpose and omits definitions of ILEFT, IMAIN, and IRIGHT.
FRESNEL	OK	Header lacks discussion of purpose and omits definitions of all inputs and outputs
SEDIFF	OK	Header does not contain subroutine purpose. Inadequate explanation of subroutine processes.
AIRY	OK	Subroutine purpose is omitted. No explanation of subroutine process is given.

Table 2.13-13 Code Quality and Internal Documentation Results (Continued)

SUBROUTINE	CODE QUALITY	INTERNAL DOCUMENTATION
CONECT	OK	Internal documentation lacks purpose of subroutine and any explanation of process. Explanations are especially necessary for limitations on the real and imaginary parts of ZETA.
GAUSSQ	OK	Header omits subroutine purpose. No explanation of subroutine processes is given.
POWERS	Dimension of arrays GCOEFF and HCOEFF could be reduced from 35 to 22	Internal documentation is non-existent.
LINFIT	OK	Header lacks discussion of purpose and output. Inadequate explanation of subroutine algorithms is given.
PARFIT	OK	Header lacks discussion of purpose and output. Inadequate explanation of subroutine algorithms is given.
SAVRST	Instead of assigning variables PSIRCS and THTRCS their own values in both save and restore modes, a comment could be made that they are unchanged.	OK

2.13.4 Software Test Cases and Results

Software testing was performed primarily by developing off-line drivers to exercise the subroutine(s) involved. Some tests were performed by running the entire ALARM 3.0 model in debug mode, or in standard mode using diagnostic write statements to examine critical variables. The sample input files used in some test cases are those that are delivered with the ALARM 3.0 code.

Decision Logic Subroutines: Seventeen tests were performed on the main logic routine LAPROP. All of these tests were combinations of numerical checks and logic checks. Every branch of the code displayed in figure 2.13-2 of the ASP II was exercised. The only numerical checks made were to verify that F^4 was calculated using the correct proportions between F_M, F_K, and F_S, as in equations (2.13-4) and (2.13-7) of the ASP II. All other numerical calculations are performed by the subroutines that LAPROP calls. Since these routines are tested separately (see below), the calls to SEDIFF, KEDIFF, and MLTPTH were commented out, and values for F_M, F_K, and F_S were hardwired in the off-line driver. All other subroutine calls in LAPROP were executed. When testing LAPROP, the called routine PROFIL was not allowed to call VISBLE; terrain elevations were hard coded.

Unless otherwise specified, each test case for decision logic was run using an off-line driver and used the values in table 2.13-14.

Table 2.13-14 Parameter Values

VARIABLE	VALUE	DEFINITION
DELTAG	90m	Ground range increment between terrain points
GRANGT	10,000m	Ground range from site to target
NPROFL	111	Number of points in terrain profile between site and target
HAMMSL	10m	Radar antenna height (MSL)
ELVSML(I)	0.1*RAN(I)	MSL height of points in terrain profile

One test was performed for subroutine FIRST and two were performed for subroutine SECOND. FIRST calculates the clearance ratio that is used by LAPROP to determine which combination of multipath and diffraction to use. SECOND calculates the "average" height to clearance ratio, which is used for the same purpose.

Subroutines PROFIL and VISBLE were tested as shown in test cases 13-21 through 13-24. PROFIL was tested for array bounds protection and correct round-off in the determination of the number of profile points between the radar and target (or jammer.) Three tests were designed for subroutine VISBLE. Two were designed to check for correct assignment of visibility. The third test was to check for a potential problem with a FORTRAN library routine when called with zero arguments.

.

Table 2.13-15 Decision Logic Software Test Cases

TEST CASE ID	TEST CASE DESCRIPTION
13-1	OBJECTIVE: Test for FTO4TH = GRBELO * GTBELO if LPPROP is false in LAPROP. PROCEDURE: 1. Set IFLSAM = 1. 2. Set LPPROP = FALSE. 3. Execute LAPROP 4. Stop in LAPROP at line 152. 5. Step to the next executable. 6. Record the line number. 7. Go to return. 8. Record the value of FTO4TH. VERIFY: Line number for step 6 is 512 and FTO4TH = 1. RESULT: OK
13-2	OBJECTIVE: Test for FTO4TH = 0.0 if LPPROP is false, target is masked, and IFLSAM = 1 in LAPROP. PROCEDURE: 1. Repeat steps 1-2 from test 13-1. 2. Set ELVMSL(50) = 300 and ZTPROF = 10. 3. Execute LAPROP. 4. Go to return. 5. Record the value of FTO4TH. VERIFY: FTO4TH is zero and target is masked. RESULT: OK
13-3	OBJECTIVE: Test for FTO4TH = 0.0 if LPPROP is false, target is masked, and IFLSAM = 0 in LAPROP. PROCEDURE: 1. Set LPPROP = False. 2. Set ELVMSL(50) = 300 and ZTPROF = 10. 3. Set IFLSAM = 0. 4. Step through the LAPROP subroutine. 5. Record the value of FT04TH. VERIFY: Lines 529-531 are executed and FTO4th is zero. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-4	OBJECTIVE: Test for spherical earth diffraction only in LAPROP. PROCEDURE: 1. Set ZTPROF = 10.0m. 2. Execute LAPROP. 3. Stop at line 224. 4. Record FSUBS and FPPROP. 5. Go to return. 6. Record FTO4TH. VERIFY: FTO4TH = $FSUBS^4$ RESULT: OK
13-5	OBJECTIVE: Test for weighted average of knife edge and spherical earth diffraction in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 17. 2. Set ZTPROF = 50m. 3. Execute LAPROP. 4. Stop at line 247 in LAPROP. 5. Record FSUBK, FSUBS, and ALFAKS. 6. Go to return. 7. Record FTO4TH. VERIFY: FTO4TH = $(\alpha FSUBK + (1-\alpha)FSUBS)^4$ where α = ALFAKS $\approx 0.0917/0.25$ RESULT: OK
13-6	OBJECTIVE: Test for knife edge diffraction only in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 34. 2. Set ZTPROF = 60m. 3. Execute LAPROP. 4. Stop at line 313. 5. Record FPPROP and FSUBK. 6. Go to return. 7. Record FTO4TH. VERIFY: FTO4TH = $FSUBK^4$. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-7	OBJECTIVE: Test for weighted average of multipath and diffraction losses. PROCEDURE: 1. Set ELVMSL(50) = 17. 2. Set ZTPROF = 120m. 3. Execute LAPROP. 4. Stop at line 389. 5. Record FSUBK, FSUBS, FSUBM, ALFAKS, ALFAMD, FSUBD, and FPPROP. 6. Go to return. 7. Record FTO4TH. VERIFY: $FSUBD = \alpha_s FSUBK + (1-\alpha_s)FSUBS$, $FPPROP = \alpha_D FSUBM + (1-\alpha_D)FSUBD$, and $FTO4TH = FPPROP^4$ RESULT: OK
13-8	OBJECTIVE: Test for pure multipath, no diffraction in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 17. 2. Set ZTPROF = 200. 3. Execute LAPROP. 4. Stop at line 471. 5. Record FSUBM and FPPROP. 6. Go to return . 7. Record FTO4TH. VERIFY: $FTO4TH = FSUBM^4$. RESULT: OK
13-9	OBJECTIVE: Test for combination of multipath and knife edge diffraction in LAPROP. PROCEDURE: 1. Set DELTAG = 120m, and ZTPROF = 140. 2. Set ELVMSL(50) = 34. 3. Execute LAPROP. 4. Stop at line 457. 5. Record FSUBK, FSUBM, ALFAMD, and FPPROP. 6. Go to return. 7. Record FTO4TH. VERIFY: $FTO4TH = FPPROP^4$ and $FPPROP = \alpha FSUBM + (1-\alpha)FSUBK$ RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-10	OBJECTIVE: Test for combination of multipath and spherical earth diffraction in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 12. 2. Set ZTPROF = 90m. 3. Execute LAPROP. 4. Stop at line 356. 5. Record FSUBS, FSUBM, ALFAMD, and FPPROP. 6. Go to return. 7. Record FTO4TH. VERIFY: $FTO4TH = FPPROP^4$ and $FPPROP = \alpha FSUBM + (1-\alpha)FSUBS$. RESULT: OK
13-11	OBJECTIVE: Test for condition where only spherical earth diffraction is called for, but it does not converge (target is not masked) in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 1.0. 2. Set ZTPROF = 10m. 3. Set CONVRG = FALSE. 4. Execute LAPROP. 5. Stop at line 267. 6. Record FSUBM and FPPROP. 7. Go to return. 8. Record FTO4TH. VERIFY: $FTO4TH = FSUBM^4$. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-12	OBJECTIVE: Test for condition where knife edge and spherical earth diffraction are called for, but spherical does not converge in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 17.0. 2. Set ZTPROF = 50m. 3. Set CONVRG = FALSE. 4. Execute LAPROP. 5. Go to line 297. 6. Record FSUBK and FPPROP. 7. Go to return. 8. Record FTO4TH. VERIFY: FTO4TH = FSUBK^4. RESULT: OK
13-13	OBJECTIVE: Test for condition where only spherical earth diffraction is called for, but it does not converge (target masked) in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 12. 2. Set ZTPROF = 7 and HAMMSL = 7m. 3. Set CONVRG = FALSE. 4. Execute LAPROP. 5. Go to line 281. 6. Record FSUBK and FPPROP. 7. Go to return. 8. Record FTO4TH. VERIFY: FTO4TH = FSUBK^4. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-14	OBJECTIVE: Test for condition where spherical earth diffraction and multipath are called for, but spherical does not converge in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 12. 2. Set ZTPROF = 70m. 3. Set CONVRG = FALSE. 4. Execute LAPROP. 5. Go to line 406. 6. Record FSUBM and FPPROP. 7. Go to return. 8. Record FTO4TH. VERIFY: FTO4TH = $FSUBM^4$. RESULT: OK
13-15	OBJECTIVE: Test for condition where all three effects are called for but spherical earth diffraction does not converge in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 24. 2. Set ZTPROF = 120m. 3. Set CONVRG = FALSE. 4. Execute LAPROP. 5. Go to line 430. 6. Record FSUBM, FSUBK, ALFAMD and FPPROP. 7. Go to return. 8. Record FTO4TH. VERIFY: FTO4TH = $[\alpha FSUBM + (1-\alpha)FSUBK]^4$. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-16	OBJECTIVE: Test calculation of FT04TH when NPROFL = 0 and receiver does not equal transmitter in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 24. 2. Set ZTPROF = 120m. 3. Set LPPROP = TRUE. 4. Set IFLSAM = 0. 5. Execute LAPROP. 6. Stop at line 499. 7. Record GTBELO and GRBELO. 8. Go to return. 9. Record FTO4TH. VERIFY: FTO4TH = GTBELO * GRBELO. RESULT: OK
13-17	OBJECTIVE: Test calculation of FTO4TH when NPROFL = 0 and receiver equals transmitter in LAPROP. PROCEDURE: 1. Set ELVMSL(50) = 24. 2. Set ZTPROF = 120m. 3. Set LPPROP = TRUE. 4. Set IFLSAM = 1. 5. Execute LAPROP. 6. Stop at line 493. 7. Record GTBELO. 8. Go to return. 9. Record FTO4TH. VERIFY: FTO4TH = $GTBELO^2$. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-18	OBJECTIVE: Test calculation of clearance ratios in FIRST. PROCEDURE: 1. Create terrain profile as follows: ZPROFL(I) = MOD(I,100) - MOD(I,200)/2, that is, two primary peaks at I=99 and I=299. 2. Set RLAMDA = 1.0μs and ZTPROF = 100m. 3. Independently calculate TANEPT, DRATIO, and FRCMIN. 4. Run off-line driver to call FIRST. 5. Record FRCMIN and DRATIO(I) for all i. VERIFY: FRCMIN and DRATIO match independent calculations. RESULT: OK
13-19	OBJECTIVE: Test the calculation of h_M in SECOND. PROCEDURE: 1. Initialize RLAMDA = 1.0μs. 2. Set ELVMSL(I) = MOD(I,100) - MOD(I,200)/2. 3. Run off-line driver to call SECOND. 4. Record HMDZRO. 5. Independently calculate h_M. VERIFY: HMDZRO matches independent calculation of h_M. RESULT: OK
13-20	OBJECTIVE: Check for agreement between Δ_0 calculated in subroutines FIRST in SECOND. PROCEDURE: 1. Run ALARM with VMS debug, using Sample 8 as the input file. 2. In FIRST, note the values of the following variables when DRATIO attains its minimum value: DSUB1, XTPROF, INDXFC, DSUB2, DELZRO 3. Note the value of the following variables in SECOND: GRRTOT, GRANGT, CONST 4. Compare the following values from first and second for flight path points 3 and 9: DSUB1 to GRRTOT, XTPROF to GRANGT DSUB2 to GRRTOT - GRANGT, DELZRO to CONST VERIFY: Variable values match in step 4. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-21	OBJECTIVE: Test integer round off and array bounds protection in PROFIL. PROCEDURE: 1. Use an off-line driver to call PROFIL (with the call to VISBLE disabled) seven times. 2. Set DELTAG to 90, GRANGT to the values in the first column below and independently calculate the values in the second column for NPROFL. GRANGT NPROFL --- 70 0 100 1 179 1 181 1 269 2 271 2 62000 >MPROFL VERIFY: Independently calculated values for NPROFL match those calculated by the PROFIL routine. RESULT: OK
13-22	OBJECTIVE: Test that every profile point is visible for a terrain that gently slopes away from the radar in VISBLE. PROCEDURE: 1. Set HAMMSL = 110m, AZIN = π/2, SPLAT = π/4, and SITLAM = π/6. 2. Execute VISBLE. 3. Record VISIBL(I) for I=1 to NPROFL. VERIFY: VISIBL(I) = TRUE for all I. RESULT: OK
13-23	OBJECTIVE: Verify that every other point is masked for a sawtooth terrain in VISBLE. PROCEDURE: 1. Set HAMMSL = 110m, AZIN = π/2, SPLAT = π/4 , SITLAM = π/6. 2. Set ELVMSL(I) = 100 - 100 * MOD(I,2). 3. Execute VISBLE. 4. Record VISIBL(I) for I=1 to NPROFL. VERIFY: VISIBL(I) = TRUE for I even and VISIBL(I) = FALSE for I odd. RESULT: OK

Table 2.13-15 Decision Logic Software Test Cases (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-24	OBJECTIVE: Check for possible problem with zero arguments in DATAN2 (noted in desk checking) in VISBLE. PROCEDURE: 1. Set HAMMSL = 110, DELTAG = 100m, GRANGT = 180000m, AZIN = 0.0, SPLAT = π/2 * 0.999, and SITLAM = π/6. 2. Set ELVMSL(I) = 0.0 for all I. 3. Execute VISIBLE. 4. Record STPHII and TERPHI at line 167 when I=100. 5. Examine the arguments of DATAN2 at line 176. 6. Record the value of TERLAM. VERIFY: STPHII = 1.0, TERPHI = π/2, TERLAM = 2π, and that the arguments of DATAN2 are zero. RESULT: OK

Specular Multipath Subroutines: Ten tests were performed to verify the design elements in MLTPTH. Table 2.13-16 contains descriptions of these tests. Three of the tests were used for numerical checks on critical calculations (13-24, 27, and 32). Two were designed to check for reasonableness of response (13-26 and 31). The remainder were used to test access to various branches of the code.

One test was designed to exercise the logical branches of subroutine RFLECT. Simultaneous checks were made on numerical accuracy for several incidence angles.

Table 2.13-16 Software Test Cases for Specular Multipath

TEST CASE ID	TEST CASE DESCRIPTION
13-25	OBJECTIVE: Check calculations of F_M in MLTPTH. PROCEDURE: 1. Use an off-line driver to execute subroutine MLTPTH. 2. Set HT = 100m, HR = 100m, RROUGH = 1.0, RT = 10000m, RLAMDA = 1.0. 3. Define an elliptical profile: ZPROFL(I) = ZPROFL(I-1) + 2 * (ZMID -ZPROFL(I-1)) where ZMID = $HT - \sqrt{b^2 - \frac{\left(\frac{RT}{2}\right)^2}{\frac{a^2}{b^2}}}$, and $a^2 = 5100^2$ and $b^2 = 1010$. 4. Stop at line 501, record DELTAR. 5. Stop at line 528, record ZONWID. 6. Stop at line 603, record DELMAX, RROUGH, RCOEFF, and FNZMAX. 7. Stop at line 629, record SUMWID, CRFLCT, FSUBM, and NAREAS. 8. Redefine the terrain profile as in step 2, but set every third value of ZPROFL to zero. 9. Repeat steps 3-6. VERIFY: 1. Most of the facets contain specular points for the first terrain profile. 2. NAREAS = 1 for the first terrain profile. 3. Most of the facets do not contain specular points for the second terrain profile. 4. NAREAS is a large number for the second terrain profile. 5. FSUBM is approximately the same for both terrain profiles. 6. FSUBM and selected values of ZONWID and CRFLCT match independent calculations. RESULT: OK

Table 2.13-16 Software Test Cases for Specular Multipath (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-26	OBJECTIVE: To exercise the portion of the code that treats the special case with a specular point in only the first facet of the terrain profile in MLTPTH. PROCEDURE: 1. Use an off-line driver to execute MLTPTH. 2. Set HT = 100m, HR = 10m, RT = 10000m, RLAMDA = 1.0m and PULWID = 10μs. 3. Let ZPROFL(0) = 9.0 ZPROFL(1) = -2.0 ZPROFL(I) = -12.0 * I XPROFL(I) = 90.0 * I, for I=2 to NPROFL 4. In subroutine MLTPTH at line 646 write out the values of NAREAS and ISTART(1). 5. At line 754 write out the values of SPCULR and DELTAR. At line 818 write out the values of FSUBM and GBELOD. 6. Using the debugger run the driver and stop in MLTPTH at line 786. Examine the value of FZNMAX and verify that FZNMAX = 0.0. VERIFY: 1. Only the terrain facet between I=0 and I=1 contains a specular point. 2. SPCULR = .TRUE. at line 754. 3. FSUBM = 1.0, because the multipath contribution is 0, due to code error in calculating ASUBS based on FZNMAX. RESULT: Error D5 found in desk checking confirmed.
13-27	OBJECTIVE: Check multipath effect as a function of altitude in MLTPTH. PROCEDURE: 1. Use an off-line driver to execute MLTPTH. 2. Set HTMN = 500m, HTMX = 750m, DELT = 10m, HR = 18.3m, RT = 27005m, RLAMDA = 1.0m, and PULWID = 10μs. 3. Let ZPROFL(I) = 0.0 and XPROFL(I) = 90.0 * I, for I=0 to NPROFL 4. Increment HT by DELT from HTMN to HTMX. For each HT recalculate FSUBM by calling MLTPTH. After each call to MLTPTH write the value of HT and $FSUBM^2$ to a file. VERIFY: $FSUBM^2$ increases with altitude. RESULT: OK

Table 2.13-16 Software Test Cases for Specular Multipath (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-28	OBJECTIVE: Numerical and logical check for roughness over sea in MLTPTH. PROCEDURE: 1. Use an off-line driver to execute MLTPTH and SEKINT. 2. Initialize LAMBDA = 1, WNDKNO = 10, LCOVER = 7, HT = 100m, HR = 10m, RT = 10,000m, and PULWID = 1.04μs. 3. Set up a terrain profile with XPROFL(I) = 90 * I and ZPROFL(I) = XPROFL(I)/10^6. 4. Stop in SEKINT at line 180 and record the values of SIGWAV and RROUGH. 5. Stop in subroutine MLTPTH at line 588 and record values of SINGAM and ASUBS. 6. As suggested by desk checking, change line 177 in SEKINT to SIGWAV = $\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{WINDSP^{2.5}}{8.67}\right)$ 7. Repeat steps 1-5. VERIFY: Real part of ASUBS matches off-line calculations for both cases. Also check difference between the two. RESULT: OK
13-29	OBJECTIVE: Check for logic cases with no terrain areas visible to both radar and target in MLTPTH. PROCEDURE: 1. Use an off-line driver to execute MLTPTH. 2. Set HT = 100m, HR = 10m, RT = 10000m, RLAMDA = 1.0m, and PULWID = 10μs. 3. Set ZPROFL(0) = 0.0 ZPROFL(1) = -2.0. ZPROFL(I) = I, for I=2 to NPROFL. 4. Print out VISIBL(I) for I=1 to NPROFL. VERIFY: 1. VISIBL(I) = False for all I except I=111=NPROFL. 2. NAREAS = 0. RESULT: OK

Table 2.13-16 Software Test Cases for Specular Multipath (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-30	OBJECTIVE: Check that a flat terrain gives same FSUBM independent of the step size in the terrain profile in MLTPTH. PROCEDURE: 1. Set up two terrain profiles, one with spacing of 90m and another with spacing of 3,000m. 2. Set HT = 100m, HR = 10m, RT = 10000m, RLAMDA = 1.0m, and PULWID = 10μs. 3. Let ZPROFL(I) = 0.0, for I=0 to NPROFL, for both terrain profiles. 4. Use an off-line driver to execute MLTPTH for both profiles. 5. Examine values of FSUBM. VERIFY: FSUBM is the same for both profiles. RESULT: There was a divide by zero error for this test in MLTPTH at line 690, because there is no protection for the case of PSLOPE = 0.0. Thus, a different type of terrain profile had to be used to test the terrain resolution effect (see test 13-31).
13-31	OBJECTIVE: Test for dependence of FSUBM on length of terrain facets in MLTPTH. PROCEDURE: 1. Set HT = 100m, RT = 10000m, HR = 10m, and RLAMDA = 1.0μs. 2. Set up terrain profile with XPROFL(I) = 90 * I and ZPROFL(I) = XPROFL(I)/10^6. 3. Use an off-line driver to execute MLTPTH. 4. Record ZONWID and FSUBM. 5. Set up a second terrain profile with XPROFL(I) = 3000 * I and ZPROFL(I) = XPROFL(I)/1000. 6. Repeat steps 3 and 4. VERIFY: FSUBM and ZONWID in step 4 are the same in both cases. RESULT: OK
13-32	OBJECTIVE: Test to compare one area versus two area multipath in MLTPTH. PROCEDURE: 1. Set HT = 10m, RT = 10001m, HT = 100m, and RLAMDA = 1.0μs. 2. For the first terrain profile: Let ZPROFL(I) = 9.0 * (55-I)/55, for I=1 to 55 ZPROFL(I) = 18.0 * (I-57)/55, for I=56 to NPROFL 3. Set up the second terrain profile the same as the first, but set ZPROFL(56) = -0.5. 4. For each terrain profile run MLTPTH in debug mode using an off-line driver. 5. Stop at line 314 and examine the value of NAREAS. VERIFY: NAREAS = 1 for the first terrain profile and that NAREAS = 2 for the second terrain profile. RESULT: OK

Table 2.13-16 Software Test Cases for Specular Multipath (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-33	OBJECTIVE: Perform numerical checks on zone width calculations in MLTPTH. PROCEDURE: 1. Set HT = 100m, HR = 10m, RT = 10000m, RLAMDA = 1.0m, PULWID = 10μs. 2. Let XPROFL(I) = 90.0 * I ZPROFL(I) = 0.000001 * XPROFL(I), for I=0 to NPROFL. 3. Use an off-line driver to execute MLTPTH. 4. Stop at line 528 in routine MLTPTH, examine DELTAR, GAMMA, RSMALL, HSUB1, HSUB2, RDELTA, AMINUS, BCOEFF, CTERM, COEFF, RADICL, and ZONWID. 5. Develop an independent PC program that calculates ZONWID. 6. In the PC program examine variables equivalent to those in step 4. VERIFY: That the values given from the MLTPTH routine equal the values given from the PC program. RESULT: OK
13-34	OBJECTIVE: Check accessibility of code segments for special cases in MLTPTH. PROCEDURE: 1. Set RROUGH = 0. 2. Use off-line driver to execute MLTPTH in debug mode. 3. Stop at call to OFFBOR, examine the values of SUMWID and NAREAS. 4. Stop at line 643 and examine the value of FSUBM. 5. Continue execution. VERIFY: 1. The next two lines of code executed are line 645 and 825. 2. SUMWID = 0.0. 3. NAREAS = 0. 4. FSUBM = 1.0. RESULT: OK

Table 2.13-16 Software Test Cases for Specular Multipath (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-35	OBJECTIVE: Check numerical accuracy and branch accessibility for subroutine RFLECT. PROCEDURE: 1. Set IPOLAR = 0, GAMMA = 0.1, and YSQUAR = (10.0, -6.0D-02). 2. Run RFLECT using an off-line driver in debug mode. 3. Stop in RFLECT after line 126 and record ESINEG, CNUMER, CDENOM, RADICL. 4. Stop after line 143 and record RCOEFF. 5. Repeat steps 1-3 with GAMMA = 0.01. 6. Repeat steps 1-3 with GAMMA = 0.001. 7. Repeat steps 1-6 with IPOLAR = 1, except in step 2 stop after line 135 and write out CNUMER, CDENOM, and RADICL. VERIFY: The value for RCOEFF in step 3 matches independent calculations for each variation of GAMMA and IPOLAR. RESULT: OK

Knife Edge Diffraction Subroutines: Three tests were designed for subroutine KEDIFF, as shown in table 2.13-17. The first test was designed to exercise the portion of the code that traps an error condition that could result in addresses beyond the array boundaries. The second test was designed to exercise the branches of the code that recognize the order of the diffraction points. There are four sub-conditions, corresponding to the four possible combinations listed in the table. The third test was designed to access a special case branch of the code.

Six tests were performed on subroutine DEYGOU, which actually calculates the F_k factor. The first test was designed to check for reasonable response to varying wavelength. Two of the tests (13-40 and 13-44) were used to check accessibility of the logical branches of the subroutine. Tests 13-41 and 43 were designed for numerical checks. One special test (13-42) was designed to verify correct inputs to subroutines TGAIN and OFFBOR, which are not part of this FE.

Three tests were performed on subroutine FRESNL. The first test was to verify the numerical output. The second test verifies that the functions are odd. The third test checks for an underflow potential identified during desk checking.

Table 2.13-17 Software Test Cases for Knife Edge Diffraction

TEST CASE ID	TEST CASE DESCRIPTION
13-36	OBJECTIVE: Check for array bounds protection in KEDIFF. PROCEDURE: Using an off-line driver, call KEDIFF with a terrain profile containing more than 128 local maxima. VERIFY: The program stops and the correct warning message occurs. RESULT: OK
13-37	OBJECTIVE: Exercise all branches that find the principle edges in KEDIFF. PROCEDURE: 1. Set up profiles with the following combinations of Knife Edges 13-37a: Main and Right 13-37b: Left and Main 13-37c: Left, Main, and Right 13-37d: Main only 2. Compute DRATIO for entire profile in each of the 4 cases. VERIFY: 1. The maxima occur at the correct ZPROFL points, and 2. The knife edges are in the correct order. RESULT: Will not declare first terrain point nor last two terrain points as knife edges, even though these could, in fact, be knife edges.
13-38	OBJECTIVE: Test branch of code that traps the "no local maxima" condition in KEDIFF. PROCEDURE: Set up terrain profile with every other point at the same height: ZPROFL(i) = MOD(i,2). VERIFY: NLOCAL = 0. RESULT: OK
13-39	OBJECTIVE: Check that F_K is sensitive to wavelength in DEYGOU. PROCEDURE: 1. Use an off-line driver to run KEDIFF and DEYGOU with a fixed profile with a single local maximum. 2. Vary λ from 0.2mm to 10m. 3. Record the values of F_K. VERIFY: F_K decreases as λ increases. RESULT: OK

Table 2.13-17 Software Test Cases for Knife Edge Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-40	OBJECTIVE: Check access to the branch of code that treats negative clearance ratios in DEYGOU. PROCEDURE: Set up geometry with the target masked, in the following steps. 1. Set target height = 385m, ground range = 35 km, and a terrain maximum height of 500m at 31km. 2. Run DEYGOU using off-line driver. 3. At line 384 write TANEPS and EPSLNP whenever DRATIO(INDXFC) ≤ 0.0. VERIFY: TANEPS and EPSLNP were written to output file. RESULT: OK
13-41	OBJECTIVE: Check numerical calculations of FLEFT, FMAIN, and FRIGHT in DEYGOU. PROCEDURE: 1. Run off-line driver for KEDIFF and DEYGOU using terrain profiles a, b, and c from test 13-37. 2. Print out the values of the clearance ratios and diffraction factors that occur for each terrain profile: a. RATIOM, FMAIN, and RATIOR, FRIGHT b. RATIOL, FLEFT, and RATIOM, FMAIN c. RATIOL, FLEFT, and RATIOM, FMAIN 3. Independently calculate these same variables. VERIFY: Clearance ratios and diffraction factors agree with independent calculations. RESULT: OK
13-42	OBJECTIVE: Check for proper input to TGAIN and OFFBORE in DEYGOU. PROCEDURE: 1. Run off-line driver for KEDIFF and DEYGOU. 2. Stop in subroutine DEYGOU before the call to OFFBOR. 3. Record values of ALPHAR, ALPHAT, EPSLNP, and EPSLNR. 4. Stop after the call to TGAIN. 5. Record OFFAZT and OFFELT. VERIFY: ALPHAR, ALPHAT, and EPSLNR are all equal to 0.0; OFFAZT = 0.0 and that OFFELT = EPSLNP. RESULT: OK

Table 2.13-17 Software Test Cases for Knife Edge Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-43	OBJECTIVE: Check adjustments to clearance ratios for secondary diffraction points in DEYGOU. PROCEDURE: 1. Run the off-line driver for KEDIFF using the conditions of test 13-37a and b. 2. Record values for a. RATIOR b. RATIOL 3. Independently calculate RATIOR and RATIOL. VERIFY: Ratios match independent calculations. RESULT: OK
13-44	OBJECTIVE: Check that F_k calculations are bypassed when DRATIO(IMAIN) > 100 in DEYGOU. PROCEDURE: 1. Set RT=35000 and HT=35000. 2. Run KEDIFF and DEYGOU with off-line driver. 3. Note value of RATIOM at line 200. 4. Continue execution. VERIFY: RATIOM > 100 and program control skips to line 249. RESULT: OK
13-45	OBJECTIVE: Numerical check of output from subroutine FRESNL. PROCEDURE: 1. Run FRESNL using an off-line driver with XARGMT = 0.0,0.1,.....,6.9. 2. Record values for COSINT AND SININT. 3. Independently calculate values of Fresnel sine and cosine functions. VERIFY: Values from FRESNL match published values and independent calculations. RESULT: OK
13-46	OBJECTIVE: Verify that the implementations in FRESNEL of both sine and cosine integrals are odd functions. PROCEDURE: 1. Run FRESNL using off-line driver with XARGMT = -0.1, -0.2,...,-0.9. 2. Record values for COSINT and SININT. 3. Compare with values for XARGMT = 0.1, 0.2,...,0.9 calculated in test 13-45. VERIFY: COSINT(-x) = - COSINT(x) and SININT(-x) = - SININT(x) for all values of x. RESULT: OK

Table 2.13-17 Software Test Cases for Knife Edge Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-47	OBJECTIVE: Check for possible underflow with small input values in FRESNL. PROCEDURE: 1. Run off-line driver with XARGMT = 10^{-9}, 10^{-10},....., 10^{-30}. 2. Record result. VERIFY: Underflow protected. RESULT: OK

Spherical Earth Diffraction Subroutines: Six tests were performed to verify the spherical earth diffraction design elements. These tests were performed by running ALARM in debug mode. Two tests were performed on subroutine SEDIFF; one to test the computation of the value of Fock's series, and one to test normalization calculations.

One test case was designed to test whether subroutine AIRY calls the correct subroutine to calculate $\overline{Ai}(z)$ for various values of z in each of the three regions of the complex plane.

Subroutine CONECT was also tested for calling the appropriate subroutine, based on regions of the complex plane containing Z1 and Z2.

One test case was used to test the calculations on subroutine GAUSSQ, and subroutine POWERS was tested to check its calculations.

.

Table 2.13-18 Software Test Cases for Spherical Earth Diffraction

TEST CASE ID	TEST CASE DESCRIPTION
13-48	OBJECTIVE: Test that SEDIFF generates the correct output. PROCEDURE: 1. Modify the SEDIFF subroutine to print out the following variables: RKFACT, CCOEFF, CONST3, CONST4, GRANGT, INDXFC, ZPROFL(INDXFC), XPROFL(INDXFC), EPSLNT, HAMMSL, HTMMSL, RLAMDA, REARTH, DRATIO(INDXFC), APARA0, APARA1, APARA2, GTILDE, CONVRG, FSUBS, and the values of J, AIRY(ZNEXPY), and AIRY(ZNEXPZ) for each iteration in the loop on J that determines convergence. 2. Run ALARM with VMS debug and use Sample 4 as the input file. Only seven target positions need to be processed. 3. Print the values of the variables listed in step 1 for the first seven target positions. 4. Use the values for the following variables from the output file for the first target position to independently determine values for CONVRG, J (number of iterations), and F_S: GRANGT, ZPROFL(INDXFC), XPROFL(INDXFC), EPSLNT, HAMMSL, HTMMSL, RLAMDA, DRATIO(INDXFC), APARA0, APARA1, APARA2, ALINE0, ALINE1, and GTILDE. 5. Repeat step 4 for the second, third, and seventh target positions. VERIFY: The values for CONVRG, J (number of iterations), and F_S generated by ALARM are the same as those generated independently. RESULT: OK

Table 2.13-18 Software Test Cases for Spherical Earth Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-49	OBJECTIVE: Test that SEDIFF correctly calculates intermediate values for both RADEFF>0 and RADEFF<0. PROCEDURE: 1. Modify the SEDIFF subroutine to print the following values: APARA0, APARA1, APARA2, REARTH, RKFACT, RADINV, RADEFF, Z1EFF, Z2EFF, HAMMSL, HTMMSL, ALINE0, ALINE1, CROOTR, RZERO, HZERO, XARGMT, YARGMT, and ZARGMT. 2. Run ALARM using Sample 3 as the input file. Only one target position needs to be processed. This will give RADEFF > 0. 3. Compare values of variables in step 1 generated by SEDIFF to independently calculated values. 4. Write an off-line driver for the instrumented version of SEDIFF described in step 1. Run using the following input values (note that the values of ELVMSL(I) should yield RADEFF < 0): HAMMSL = 282 HTMMSL = 369 GRANGT = 101 DELTAG = 10 RLAMDA = .226 XPROFL(I) = DELTAG*I $ELVMSL(I) = [XPROFL(I)-50]^2$ 5. Compare the values of the variables in step 1 generated by SEDIFF to independently calculated values. VERIFY: Values generated by SEDIFF match independently calculated values. RESULT: OK

Table 2.13-18 Software Test Cases for Spherical Earth Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-50	OBJECTIVE: Test that AIRY calls the correct subroutine. PROCEDURE: 1. Modify AIRY to read, print, and use a new value of ZARGMT. 2. Run ALARM using VMS debug with breaks at the entry points for: CONECT, GAUSSQ, and POWERS. Do this for each of the following values of ZARGMT to be read by AIRY: 1. (-2.0, +2.0) 2. (-2.0, -2.0) 3. (2.0, +2.0) 4. (2.0, -2.0) 5. (-1.0, -3.0) 6. (-1.0, +3.0) 7. (-1.0, +1.0) 8. (-1.0, -1.0) 9. (-1.0, +5.0) 10. (-1.0, -5.0) 11. (0.0, +4.0) 12. (0.0, +2.0) 13. (0.0, -2.0) 14. (0.0, -4.0) 15. (0.0, -5.0) 16. (0.0, +5.0) 17. (-2.0, 0.0) 18. (+2.0, 0.0) 19. (+1.0, 0.0) 20. (-2.002, +3.4631) 21. (+2.002, -3.4631) 22. (0.0, +0.0) 23. (-1.99, +3.4631) 24. (-1.99, -3.4631) 25. (-1.99, +3.6) 26. (-1.99, -3.6) 3. Record which of the three routines was called for each case and compare to predetermined regions. VERIFY: The expected subroutines were called with the correct ZARGMT value. RESULT: OK

Table 2.13-18 Software Test Cases for Spherical Earth Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-51	OBJECTIVE: Test that CONECT calls the correct subroutine. PROCEDURE: 1. Modify AIRY to read, print, and use a new value of ZARGMT. 2. Independently calculate Z1 and Z2 for the following values of ZARGMT and determine which regions contain Z1 and Z2. 1. (-2.0, +2.0) 2. (-2.0, -2.0) 3. (-2.0, 0.0) 4. (-4.0, +4.0) 5. (-4.0, -4.0) 6. (-4.0, 0.0) 7. (-9.0, +4.0) 8. (-9.0, -4.0) 9. (-9.0, 0.0) 10. (-1.0, +1.0) 11. (-1.0, -1.0) 12. (-1.0, 0.0) 13. (-0.5, +0.5) 14. (-0.5, -0.5) 15. (-0.5, 0.0) 3. Run ALARM for each of the above values of ZARGMT using VMS debug. Set breaks at the entry points of CONECT, GAUSSQ, and POWERS. VERIFY: All of the above test points call the CONECT subroutine and that each call to CONECT generates two correct calls to GAUSSQ and/or POWERS, depending on the values of Z1 and Z2. RESULT: OK

Table 2.13-18 Software Test Cases for Spherical Earth Diffraction (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-52	OBJECTIVE: Test that GAUSSQ generates the correct results. PROCEDURE: 1. Modify AIRY to read, print, and use a new value of ZARGMT. 2. Independently calculate the results of GAUSSQ for the following values of ZARGMT. 1. (+2.0, +2.0) 2. (+2.0, -2.0) 3. (0.0, +5.0) 4. (0.0, -5.0) 5. (+3.0, 0.0) 6. (-2.0, +5.0) 7. (-2.0, -5.0) 3. Run ALARM for each of the above values of ZARGMT using VMS debug. Set breaks at the entry point of GAUSSQ and at the return from GAUSSQ. VERIFY: All of the above test points call the GAUSSQ function and the values of GAUSSQ equal the precalculated values for GAUSSQ. RESULT: OK
13-53	OBJECTIVE: Test that POWERS generates the correct results. PROCEDURE: 1. Modify AIRY to read, print, and use a new value of ZARGMT. 2. Independently calculate the results of POWERS for the following values of ZARGMT. 1. (+4.0, 0.0) 2. (+4.0, +1.0) 3. (+4.0, -1.0) 4. (0.0, +5.0) 5. (0.0, -5.0) 6. (-1.0, +5.0) 7. (-1.0, -5.0) 3. Run ALARM for each of the above values of ZARGMT using VMS debug. Set breaks at the entry point of POWERS and at the return from POWERS. VERIFY: All of the above test points call the POWERS function and the values of POWERS equal the precalculated values for POWERS. RESULT: OK

Initialization and Utility Subroutines: Subroutines LINFIT and PARFIT are mathematical utility routines that are used in this FE to determine the best linear or parabolic fit (least squares) to the terrain profile data points. These were tested by comparing their results to independently calculated results. Subroutine PARFIT required three tests to consider a general case and two special cases.

One test was run to check that subroutine SAVRST interchanges radar and jammer data correctly.

The remaining test cases were for the user input and initialization routines; SEKINP reads the input data specific to this FE, SEKERR checks those inputs for errors, and SEKPRT echoes those inputs. SEKINT performs initial calculations that will apply to all target positions. SEKINP, SEKERR, and SEKINT were tested as a unit, except for one test of SEKINT. SEKPRT was tested separately.

.

Table 2.13-19 Software Test Cases for Initialization and Utilities

TEST CASE ID	TEST CASE DESCRIPTION
13-54	OBJECTIVE: Test the numerical accuracy of line fit equations in LINFIT. PROCEDURE: 1. Initialize DELTAG = 90m and NPROFL = 111. 2. Set ELVMSL(I) = MOD(I,100) - MOD(I,200)/2. 3. Independently calculate coefficients of linear fit. 4. Run off-line driver to call LINFIT. 5. Record ALINE0 and ALINE1. VERIFY: ALINE values match independent calculations. RESULT: OK

Table 2.13-19 Software Test Cases for Initialization and Utilities (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-55	OBJECTIVE: Test that PARFIT generates the correct results. PROCEDURE: 1. Use the following three sets of input data for PARFIT

VARIABLES	CASE1	CASE2	CASE3
NPROFL	5	10	25
DELTAG	90	200	90
ELVMSL(1)	26302	29082	17747
ELVMSL(2)	27105	25965	7528
ELVMSL(3)	14936	26315	13660
ELVMSL(4)	7277	9491	30551
ELVMSL(5)	13268	2413	14835
ELVMSL(6)	0	11180	8893
ELVMSL(7)	0	27024	12492
ELVMSL(8)	0	19037	17168
ELVMSL(9)	0	173	13401
ELVMSL(10)	0	32500	7019
ELVMSL(11)	0	0	30348
ELVMSL(12)	0	0	16466
ELVMSL(13)	0	0	9148
ELVMSL(14)	0	0	30706
ELVMSL(15)	0	0	31600
ELVMSL(16)	0	0	20477
ELVMSL(17)	0	0	26368
ELVMSL(18)	0	0	24421
ELVMSL(19)	0	0	24137
ELVMSL(20)	0	0	24493
ELVMSL(21)	0	0	28902
ELVMSL(22)	0	0	5256
ELVMSL(23)	0	0	24240
ELVMSL(24)	0	0	10740
ELVMSL(25)	0	0	23667

(For each case, NPROFL and DELTAG were fixed, and the terrain elevations were randomly generated.)

2. Independently calculate the coefficients for the least squares parabolic fit to the elevation data in CASE1.
3. Run ALARM using VMS debug. Set break points at the entry point of PARFIT and at the return from PARFIT. At the first breakpoint deposit the data in CASE1.
4. At the next breakpoint record the PARFIT coefficients APARA0, APARA1, and APARA2.
5. Repeat steps 2-4 for CASE2 and CASE3.

VERIFY: The values for APARA0, APARA1, and APARA2 in PARFIT are the same as those calculated in step 2.

RESULT: OK

Table 2.13-19 Software Test Cases for Initialization and Utilities (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-56	OBJECTIVE: Test that PARFIT generates the correct results when there is a single terrain profile point. PROCEDURE: 1. Independently calculate the coefficients for the least squares parabolic fit for the following values: VARIABLES — VALUES NPROFL — 1 DELTAG — 90 ELVMSL(1) — 2000 2. Run ALARM using VMS debug. Set break points at the entry point of PARFIT and at the return from PARFIT. 3. Deposit the values in step 1 at the first breakpoint. 4. At the next breakpoint record the PARFIT coefficients APARA0, APARA1, and APARA2. VERIFY: The values for APARA0, APARA1, and APARA2 match those from step 1. RESULT: OK
13-57	OBJECTIVE: Test that PARFIT generates the correct results when there are only two terrain profile points. PROCEDURE: 1. Independently calculate coefficients for the least squares parabolic fit for the following values: VARIABLES — VALUES NPROFL — 2 DELTAG — 90 ELVMSL(1) — 1000 ELVMSL(2) — 2000 2. Run ALARM using VMS debug. Set break points at the entry point of PARFIT and at the return from PARFIT. 3. Deposit the values in step 1 at the first breakpoint. 4. At the next breakpoint record the PARFIT coefficients APARA0, APARA1, and APARA2. VERIFY: The values for APARA0 and APARA1 match those from step 1. RESULT: OK

Table 2.13-19 Software Test Cases for Initialization and Utilities (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-58	OBJECTIVE: Test that SAVRST generates the correct output PROCEDURE: 1. Modify the ALARM.FOR file by adding the following segments of code. At line 97: INTEGER I,J,K DOUBLE PRECISION LAMBDA, WAVEL At line 140 J=4 I=0 LAMBDA=5.0 WAVEL=10.0 DO 224 K=1,J CALL SAVRST(I, LAMBDA, WAVEL) 224 CONTINUE 2. Run ALARM with VMS debug. Set breakpoints at the call to SAVRST and at the subroutine SAVRST. 3. When execution pauses at SAVRST set breakpoints at lines 69, 124, and 172. 4. Check that execution does not pause at lines 69 or 124. 5. When execution pauses at line 150 of ALARM deposit 3 into variable I. 6. Check that execution does not pause at lines 69 or 124 of SAVRST. 7. When execution pauses at line 150 of ALARM deposit 1 into variable I. 8. Check that execution pauses at line 69 of SAVRST. Deposit 10.0 into common variable SIAPLT, 20.0 into common variable SIPHIT, 30.0 into common variable SINXIT, and 40.0 into common variable ZTPROF. 9. When execution pauses at line 172, check that the following variables contain the indicated values: SIALPS = 10.0, SIPHIS = 20.0, SINXIS = 30.0, ZTPROF = 40.0. 10. When execution pauses at line at line 150 of ALARM deposit 2 into variable I. 11. When execution pauses at line 124 of SAVRST, deposit -10.0 into common variable SIAPLT, -20.0 into common variable SIPHIT, -30.0 into common variable SINXIT, and -40.0 into common variable ZTPROF. 12. When execution stops at line 172 of SAVRST, ensure that the following variables contain the indicated values: SIALPS = 10.0, SIPHIS = 20.0, SINXIS = 30.0, ZTPROF = 40.0. VERIFY: Expected results are given in steps 9 and 12. RESULT: OK

Table 2.13-19 Software Test Cases for Initialization and Utilities (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-59	OBJECTIVE: Test that a standard, error free, input file is correctly processed in SEKINP, SEKERR, and SEKINT. PROCEDURE: 1. Develop an off-line driver that calls SEKINP, SEKERR, and SEKINT. 2. Extract the DATASEKE block from Sample 16 and use it as the input file for the driver. 3. Initialize the following variables: FREQIN = 300.0D0, REZERO = 6.37D6, LCOVER = 7, TWOPI = 2π, PI = π, INUNIT = 16 4. Run using VMS debug. 5. Set breakpoints at the returns from SEKINP, SEKERR, and SEKINT. 6. At the first breakpoint record the values of IPRSEK, IPPROP, EPSLN1, SIGMHO, RROUGH, RKFACT, TAURLX, and WNDKNO. 7. At the next breakpoint record the value of NUMERR(13). 8. At the next breakpoint record the values of LPPROP, DELTAG, REARTH, YSQUAR, RROUGH, and the first ten values of SINBET, COSBET, and CBETAP. 9. Independently calculate the values in step 8. VERIFY: SEKINP reads the correct values from the Sample-16 input file, no errors are detected by SEKERR, and SEKINT correctly calculates the values of the variables listed in step 8. RESULT: OK
13-60	OBJECTIVE: Test that SEKERR correctly detects errors, and that all logical branches are exercised. PROCEDURE: 1. Develop an off-line driver that calls SEKERR three times. 2. Run the driver using VMS debug. 3. Set a breakpoint at SEKERR. 4. At the first breakpoint make the following deposits: IPRSEK = 2, IPPROP = 2, EPSLN1 = -1.0, SIGMHO = -1.0, RROUGH = 2.0, RKFACT = 0.0, TAURLX = -1.0, WNDKNO = -1.0. 5. At the next breakpoint deposit 9999 into ISBLOC(13). 6. At the next breakpoint deposit 1 into ISBLOC(13). VERIFY: 1. Error messages are given for each of the of the variables in step 4 and NUMERR(13) = 8. 2. A "data block not encountered" error is given in step 5 and NUMERR = 1. 3. A "premature end-of-file" error is given in step 6 and that NUMERR = 1. RESULT: OK

Table 2.13-19 Software Test Cases for Initialization and Utilities (Continued)

TEST CASE ID	TEST CASE DESCRIPTION
13-61	OBJECTIVE: Test that SEKINP outputs appropriate error messages when reading invalid data. PROCEDURE: 1. In the DATASEKE block from Sample 16, change the value of IPRSEK to 1.1 and the value of EPSLN1 to 'A'. 2. Develop an off-line driver to run SEKINP and RDSTAT. 3. Use the data created in step 1 as the data input file 4. Run the off-line driver. VERIFY: A "format conversion" error is given for each record. RESULT: OK
13-62	OBJECTIVE: Test that SEKINT correctly determines the initial terrain and dielectric constants for propagation over land. PROCEDURE: 1. Develop an off-line driver to run SEKINT. 2. Initialize the following variables. IPPROP = 1, PI = 3.14, REZERO = 6.37D6, RKFACT = 3.0D0/4.0D0, EPSLN1 = 10.0D0, RLAMDA = 1.0D0, SIGMHO = 1.0D-03, LCOVER = 4. 3. Run off-line driver. 4. Set a breakpoint at the return from SEKINT. 5. At the breakpoint record the value of LPPROP, DELTAG, REARTH, and YSQUAR. 6. Independently calculate the values of the variables in step 5. VERIFY: SEKINT initializes the variables in step 5 correctly. RESULT: OK
13-63	OBJECTIVE: Test that SEKPRT generates the correct output. PROCEDURE: 1. Add a call to SEKPRT to the ALARM.FOR file. 2. Run ALARM with VMS debug and use Sample 1 as the input file. 3. Repeat step 2 with Sample 9 as input. VERIFY: The data in the output file FOR066 agree with the inputs in the DATASEKE data block of the input file. RESULT: OK

2.13.5 Conclusions and Recommendations

Code Discrepancies: In general, the multipath and diffraction FE was found to be implemented as specified in Section 2 of ASP II; however, a few discrepancies were found.

The broadest discrepancy is that ALARM 3.0 does not exactly match the MIT Lincoln Laboratory SEKE code. Apparently, the MIT model is considered to be a working tool that is not subject to rigorous coding standards, and it is changed frequently without documentation. In implementing the SEKE algorithms in ALARM, the developer changed the code in the then current version of SEKE to correct and simplify the code, bringing it into compliance with the coding standards for ALARM. At that time, the developer compared the ALARM outputs to the SEKE outputs and reported that the outputs matched. The developer also reports that Lincoln Laboratories informally approved the implementation of the SEKE algorithms in ALARM at that time. However, questions have now arisen about how well ALARM matches SEKE. The differences with SEKE are not oversights; the developer described the design as implemented. An in-depth examination of the MIT code was out of the scope of this verification effort; however, the following three items were clearly different in ALARM and MIT SEKE:

1. The constants in the logic to determine which propagation effects will be calculated.

2. The definition of the terrain profile.

3. The extent of the tangent plane used in the calculation of the multipath effect.

Apart from the differences from SEKE, several other discrepancies were also found in the Multipath and Diffraction FE. These are described in detail in Section 2.13.3. Based on the results presented in that section, the following corrections are recommended for subroutines MLTPTH, VISBLE and SEKINT:

1. In MLTPTH, remove zone width from the multipath calculation for the special case where only the first terrain point is declared specular, so that the effects are not automatically zero.

2. Modify subroutine MLTPTH to consider the possibility that the receiver gain is different from the transmitter gain.

3. Modify subroutine MLTPTH to consider the possibility that a point actually in the terrain profile is a specular point.

4. Modify MLTPTH to consider the possibility that a specular point lies in the first terrain interval even if other specular points also exist.

5. In subroutine KEDIFF, consider allowing the first point and the last two points in the terrain profile to be knife edge diffraction points, or at least discuss this issue in the documentation.

6. Modify the calculation of the terrain roughness factor in SEKINT to agree with [A.1-5].

Code Quality and Internal Documentation: The quality of the code for the Multipath and Diffraction FE in ALARM 3.0 is generally good; however, the following areas should be considered for improvement:

1. LAPROP should be examined to determine whether the logical flow could be simplified.

2. Since Δ_0 in calculated in both FIRST and SECOND, it should be included in a common so that it could be calculated only once.

3. A conditional check should be added to PROFIL so that VISBLE is not called if NPROFL ≤ 0.

4. In subroutine VISBLE, it is possible (but highly unlikely) that both arguments of DATAN2 are zero, causing the function to be undefined.

5. In subroutine MLTPTH, SLPTGL should be used instead of recalculating it as TANTRG.

6. Subroutine MLTPTH uses division by the slope between two terrain points; this presents a (slight) possibility of division by zero.

7. Subroutine SAVRST could be modified to check for disallowed values of ISVRS and to remove the code that sets PSIRCS and THTRCS to their own values.

Internal documentation is generally good, but some deficiencies were found. Subroutine headings should be standardized to include a purpose and lists of inputs and outputs for all subroutines. In addition, the comments in several subroutines need to be corrected, clarified, and/or amplified as described in the notes for the verification matrices.

External Documentation: The ALARM external documentation for this FE is inadequate. The analyst's manual basically describes only the decision logic in subroutine LAPROP and then references the MIT documentation for the Lincoln Laboratories SEKE model. This is unacceptable for two reasons: first, the ALARM implementation is not exactly the same as the SEKE implementation, and second, the MIT documentation itself is inadequate. The MIT documents describe only parts of several different versions of their model; they are inconsistent and incomplete.